Appendix F: Vendor-Specific DBMS Considerations

This appendix contains information that is DBMS vendor-specific. It will be updated as COTS versions change, or as other vendor products are supported.

F-1. Oracle RDBMS

This section contains information specific to the Oracle Database Server Segment and to database segments written for the Oracle RDBMS. It also provides information for developers of application segments that will access Oracle databases.

F-1.1 Oracle Server Segment

The Oracle RDBMS Server segment contains the software required to install and maintain an instance and a database, and the software to support both client/server and server/server communications across the network. It provides the executables for the RDBMS, and the SQL*Net network. It also contains the core database instance: Database files, tablespaces, and internal Oracle data structures (data dictionary, etc.) that are used in common by all databases within the DBMS. Finally, this segment has the tools used by database administrators to administer the DBMS and databases. Table F-1 shows the server configuration.

DBMS Executables	Oracle RDBMS, SQLDBA, Names Server, Server Manager, Export, Import, SQL*Plus, SQL*Loader
Application executables	SQL*Forms 3.0, Oracle*Forms 4.5, Oracle*Reports 2.5
Core database configuration	Tablespaces for System, Rollback Segments, Temporary, Tools and Users Log file archiving turned off
Data dictionary	Core data dictionary objects Support objects for performance monitoring Support objects for stored packages
Application database objects	Data objects for each executable Stored packages and procedures for each executable
DBA account	Operating System '`dba`' account group '`dbadmin`' account in '`dba`' group DB Admin. only No ownership or direct connection

Table F-1: Oracle RDBMS Server Configuration

DISA builds the Oracle DBMS server segment in the following manner. Oracle's installer is used to load executables. Then the core database is built based on the aggregate sizing information provided by developers. Once the database is created, the data dictionary is loaded. The data dictionary load includes the core objects required for the DBMS to function, additional items to ease DBA tasks, and packages and procedures that allow developers to use the features of Oracle 7.

The core database is installed with two accounts, `sys` (the owner of the core database) and `system` (the database administrator). Security measures mandate that the passwords of both the sys and system accounts be changed upon successful installation of this segment. The use of scripts or applications containing hard coded passwords for these accounts, or any other Oracle database accounts, is prohibited.

Log file archiving is turned off in the initial DBMS configuration. This allows applications' database segments to be installed and their data loaded without generating massive numbers of useless log files. Once a site's configuration is completely loaded, the DBA at that site must turn archive logging on to support DBMS recovery.

The Oracle RDBMS provides centralized recovery features (Rollback Segments, Online Log Files, and archived Log Files). Their configuration is set when the COTS DBMS Segment is built by DISA. Developers who need specific configurations of these components must include their requirements in their Segment Registration. Any such modifications to the COTS DBMS Segment are subject to the review and approval of the DII COE Chief Engineer and will be implemented by DISA, not the developers.

The last component of the core database segment is a model database administrator account named '`oradba`.' This is a Unix account that has the privileges needed for database administration. The '`oracle`' account that owns the executables is a special Solaris account without login capability. No user can connect to '`oracle`' directly. A user must login as a normal user, use the '`su`' command to become '`root`' and then connect to the '`oracle`' account.

F-1.2 Oracle Client Segment

The Oracle COTS RDBMS Client segment contains the Oracle RDBMS executables for to support client/server communications across the network. Depending on the needs of other segments it may also provide executables for accessing the database, tools and utilities to support data extraction, loading, and backups. The Oracle client services segment is loaded on each application server and on client workstations. Small client workstations may NFS mount the disk/directory containing the Oracle segment. It contains the runtime applications provided by Oracle (e.g., SQL*Forms 'runform') and the client portion of SQL*Net. DISA supplies different client segments for the various DII workstations to support their varying display and keyboard configurations. Table F-2 shows the client configuration.

Executables	SQL*Net (client), Oracle Forms (runform), Oracle Reports (runrep), Export/Import, SQL*Loader, SQL*Plus

Table F-2: Oracle RDBMS Client Configuration

F-1.3 Oracle Application Database Segments

There are some special considerations for mission applications that use Oracle services.

F-1.3.1 DBS_files Directory

The functional storage allocation of a Data Store is implemented in Oracle from within the DBMS as 'tablespaces.' Database segments using Oracle must create one or more tablespaces to store their objects (one tablespace for each Data Store). Segments are prohibited from storing tables and indexes in another segment's tablespace, or in the SYSTEM, TEMP, USERS, or TOOLS tablespaces. A segment may create a tablespace composed of multiple files if the storage requirements are large enough to span physical devices.

All data files for the database segment will be place in the `DBS_files` directory. The `DBS_files` directory in an Oracle database segment must be owned by '`oracle`' in group '`dba`.' The change of ownership must occur before the CREATE TABLESPACE command is issued. The DBA must issue the CREATE TABLESPACE command.

F-1.3.2 Oracle Tables

The creation of Oracle tables must specify the storage parameters for controlling their size. In addition, the create script must explicitly specify the Oracle tablespace that is to store the table. The following script excerpt creates a database table in Oracle.

```
CREATE TABLE SORTSM_BIDES (
      UIC       VARCHAR2 (6)   NOT NULL
     ,SECUR     VARCHAR2 (2)   NOT NULL
     ,TIME      DATE       NOT NULL
     ,SERV      VARCHAR2 (1)   NULL
     ,ANAME     VARCHAR2 (30)  NULL
     ,UTC       VARCHAR2 (5)   NULL
     ,ULC       VARCHAR2 (3)   NULL
     ,SCLAS     VARCHAR2 (2)   NOT NULL
     ,MJCOM     VARCHAR2 (6)   NULL
     ,MONOR     VARCHAR2 (6)   NULL
     ,MAJOR     VARCHAR2 (1)   NULL
     ,REVAL     VARCHAR2 (1)   NULL
     ,UDC       VARCHAR2 (1)   NULL
     ,LNAME     VARCHAR2 (55)  NULL
     ,COAFF     VARCHAR2 (2)   NULL
```

```
     ,BUPDATE  DATE NULL
     ,TPSN     VARCHAR2 (7)   NULL)
TABLESPACE DBSORT_DATA,
STORAGE(
     INITIAL   2000000
     NEXT      500000
     MAXEXTENTS     99
     PCTINCREASE    0);
```

F-1.3.3 Oracle Views

Views should be created after all tables have been created to guarantee that the referenced table objects exist. A segment may create views to support legacy structures as well as views to support application-specific database requirements. The following example script excerpt creates a legacy view in Oracle to support a change in a column name from SECURITY to SECUR.

```
CREATE VIEW SORTSM_BIDES_1_1
      (UIC, SECURITY, TIME, SERV, ANAME, UTC , ULC,
SCLAS , MJCOM ,
       MONOR, MAJOR, REVAL , UDC, LNAME , COAFF,
BUPDATE, TPSN)
 AS SELECT UIC, SECUR, TIME, SERV, ANAME, UTC , ULC,
SCLAS , MJCOM,
       MONOR, MAJOR, REVAL , UDC, LNAME , COAFF,
BUPDATE, TPSN
 FROM SORTSM_BIDES;
```

F-1.3.4 Oracle Constraints

When the install process includes a base load of data it can be done faster if the primary keys and indexes are created after the data is loaded. The following script excerpt creates a primary key constraint in Oracle.

```
ALTER TABLE SORTSM_BIDES ADD PRIMARY KEY (UIC)
USING INDEX TABLESPACE DBSORT_INDEX
STORAGE (INITIAL 250K NEXT 100K PCTINCREASE 0);
eof
;;
```

Note: `PCTINCREASE` should always be 0.

F-1.3.5 Oracle Packages

A database package is an Oracle feature that provides a way of grouping and storing related procedures, functions, cursors and variables together as a unit in the database for continued use. Packaged procedures and functions can be called explicitly by applications or users.

A package is dependent on all objects referenced in the constructs of its body. A package is successfully compiled only when all referenced objects are present and of the expected structure, and if the owner of the package has the necessary privileges for the referenced objects. A package becomes invalid if any of the objects referenced within the package is altered or dropped. When an application invokes a package's public procedure or function, this application must have the privilege to execute the package. Having this privilege also allows the application to execute any construct within that package.

F-1.3.6 Oracle Indexes

An Oracle index should specify a tablespace and the indexes' storage parameters. The following script excerpt creates an index in Oracle.

```
CREATE INDEX SORTSM_BIDES_LNAME
 ON DBSORT.SORTSM_BIDES (LNAME)
           TABLESPACE DBSORT_INDEX
 STORAGE (INITIAL 250K NEXT 100K PCTINCREASE 0);
eof
;;
```

F-1.3.7 Oracle Grants

Grants assign privileges on objects to database roles. In Oracle the following object-level privileges are available:

- **Select:** The object may be queried
- **Insert:** Records may be added to the object
- **Update:** Existing records in the object may be changed
- **Delete:** Existing records in the object may be deleted
- **Reference:** A table may be used (referenced) to validate a foreign key or domain key constraint

Two other object-level privileges exist in Oracle. They shall not be granted to roles or to users. These privileges are:

- **Index:** An index may be created on the object
- **Alter:** The object may be altered. Columns can be added or deleted and their definitions can be modified. Constraints can be created, modified, or deleted. This grant implicitly grants the right to delete the object as well.

The following script excerpt performs grants in Oracle.

```
sqlplus -silent DBSORT/${DBO_PWD} <<eof
GRANT DELETE,INSERT, SELECT,UPDATE ON DBSORT.
SORTSM_BIDES to DBSORT_RW;
GRANT SELECT ON DBSORT. SORTSM_BIDES to DBSORT_RO;
eof
;;
```

F-1.3.8 Oracle Database Roles

An Oracle role is a set of privileges that can be granted to users or to other roles. An Oracle user can take on multiple roles. An Oracle user can have multiple active roles or can have some roles active while others are inactive. The DBA creates roles, not the owner. Either the owner or the DBA can assign grants, as discussed above, to the role. The following script excerpt provides an example of the creation of two Oracle roles.

```
CREATE ROLE DBSORT_RO IDENTIFIED BY DBSORT_RO;
CREATE ROLE DBSORT_RW IDENTIFIED BY DBSORT_RW;
```

F-1.4 Oracle Environment Variables

Key environment variables the Oracle system uses are defined during the installation of the Oracle COTS RDBMS Server segment. These environment variables set a common environment for all users, define the Oracle system home directory, and uniquely identify the Oracle database instance.

The following environment variables are set during the installation of the Oracle COTS RDBMS Server segments.

```
ORACLE_HOME          /h/COTS/ORACLE
ORACLE_SERVER        <name of data server host>
ORACLE_SID           GCCS
TWO_TASK             T:${ORACLE_SERVER}:${ORACLE_SID}
path                 ($path:${ORACLE_HOME}/bin)
```

These environment variables are accessed through two initialization files on the application servers or client workstations. The files are named `coraenv` for C Shell and `oraenv` for Korn or Bourne Shells.

F-1.5 Oracle Synonyms

The Oracle RDBMS provides synonyms as alternate names for tables, views or other database objects. Private synonyms are user-owned aliases. Public synonyms are system wide aliases.

Developers shall not use Oracle Public Synonyms to identify data tables or views. Public Synonyms belonging to one segment may now, or in the future, conflict with object names

belonging to other applications or to the DBMS. Applications should reference objects using full object specifications (owner.table.column).

F-1.6 COE Tool Considerations

Some of the COE tools require special considerations when Oracle is used. This subsection lists the special considerations by tool.

F-1.6.1 COECreateDS

For an Oracle implementation, `COECreateDS` creates the tablespaces and the database owner account. The database owner account created by this service has an initial password equal to the account name. The account has connect privileges, and has unlimited quotas on the tablespaces created by `COECreateDS`. Its default tablespace is set to the first '`DATA`' tablespace created by the service. If no '`DATA`' tablespace is created by the service, then the first named tablespace is set as the default.

F-1.6.2 COEExtendDS

For an Oracle implementation, `COEExtendDS` adds additional storage to a tablespace. If the new storage requirements can be serviced then the tablespace is altered by allocating the disk storage to the tablespace.

F-2. Sybase RDBMS

This section contains information specific to the Sybase Database Server Segment and to database segments written for the Sybase RDBMS. It also provides information for developers of application segments that will access Sybase databases.

'sa' is the Sybase DBO for all databases. Each segment creates its own database and then grants 'create object' privileges to the segment's DBO account.

Log archiving is accomplished by the Sybase Backup Server.

F-2.1 Sybase Server Segment

The Sybase COTS RDBMS Server segment contains the software required to install and maintain an instance and a database, and the software to support both client/server and server/server communications across the network. Installation of this segment also creates the database files, a master database, log files, temporary storage, and internal data structures that make up a core database to be expanded and populated by other database segments.

The core database is installed with an `sa` account that is the owner of both the core database and the database administrator. Security measures mandate that the password of the `sa` account be changed upon successful installation of this segment. The use of scripts or applications containing hard coded passwords for these accounts, or any other Sybase database accounts, is prohibited.

The Sybase DBMS services segment contains the executables for the RDBMS, network services and the core database instance. The default owner and database administrator is the 'sa' account. The password of the `sa` account must be changed after the server segment is installed.

The Sybase server segment also sets up the initial database configuration. The initial database includes the master and model databases, and the backup server.

The Sybase 'master' and DBO databases are on mirrored disks. The Backup Server is on a separate set of mirrored disks and is also configurable for off-line archiving to 'dump device.'

Sybase provides separate Logs in each database for online recovery. Logs are archived using the Backup Server to support off-line recovery. Developers are responsible for creating the Logs in their DBO database during the segment's `PostInstall` using COE`CreateDS` or the appropriated database definition script.

F-2.2 Sybase Client Segment

The Sybase COTS RDBMS Client segment contains the Sybase RDBMS executables for accessing the database, the tools and utilities that support data storage, retrieval and

backups, and software to support client/server communications across the network. The Sybase client services segment is loaded on each application server. It contains the Sybase client executables. It also has the interfaces file.

F-2.3 Sybase Application Database Segments

There are some special considerations for mission applications that use Sybase services.

F-2.3.1 DBS_files Directory

For Sybase, segments must create a database and one or more "Sybase segments[1]" within that database for storing their objects. The database segment's prefix will be used as the Sybase database name and as the name of the DBO account. A log device must also be created for the database. It is a violation of the *I&RTS* for a segment to store its objects in another segment's database or the master database. A segment may create a database that is composed of multiple database devices if the storage requirements are large enough to span physical devices.

F-2.3.2 Sybase Database Rules

Sybase provides a functionality known as *rules*. Rules are named objects that specify the domain of acceptable values for a particular column or for any column of a custom data type. After a rule is created it can be bound to columns or custom data types.

Database rules should be created before the database tables. The binding of rules can be done when the database tables are created or when constraints are defined. The following script excerpt creates a database rule for UICs with the sample DBSORT data store segment in Sybase.

```
CREATE_RULE()
isql -UDBSORT -PDBSORT -S${DSQUERY} <<eof
create rule UIC_RULE as @UIC like '[NFWME]%'
go
eof
;;
```

F-2.3.3 Sybase Tables

Sybase tables are implicitly created within the database (schema and storage area) belonging to the database segment. If that database is split up into "Sybase partitions," the create script must explicitly specify the "Sybase partition" that is to store the table. The following script excerpt creates database tables in Sybase.

[1] Sybase uses the term *segment* to define a name to an allocated area of a database device. In this appendix, all references to a physical Sybase segment will be in quotes to differentiate it from a database segment.

```
CREATE TABLE SORTSM_BIDES (
     UIC       VARCHAR (6)    NOT NULL
    ,SECUR     VARCHAR (2)    NOT NULL
    ,TIME      DATETIME  NOT NULL
    ,SERV      VARCHAR (1)    NULL
    ,ANAME     VARCHAR (30)   NULL
    ,UTC       VARCHAR (5)    NULL
    ,ULC       VARCHAR (3)    NULL
    ,SCLAS     VARCHAR (2)    NOT NULL
    ,MJCOM     VARCHAR (6)    NULL
    ,MONOR     VARCHAR (6)    NULL
    ,MAJOR     VARCHAR (1)    NULL
    ,REVAL     VARCHAR (1)    NULL
    ,UDC       VARCHAR (1)    NULL
    ,LNAME     VARCHAR (55)   NULL
    ,COAFF     VARCHAR (2)    NULL
    ,BUPDATE   DATETIME  NULL
    ,TPSN      VARCHAR (7)    NULL)
ON DBSORT_DATA
go
eof
;;
```

F-2.3.4 Sybase Views

Views should be created after all tables have been created to guarantee that the referenced table objects exist. The following script excerpt creates database views in Sybase to support a change in a column name from SECURITY to SECUR.

```
CREATE VIEW SORTSM_BIDES_1_1
     (UIC, SECURITY, TIME, SERV, ANAME, UTC , ULC,
SCLAS , MJCOM ,
      MONOR, MAJOR, REVAL , UDC, LNAME , COAFF,
BUPDATE, TPSN)
 AS SELECT UIC, SECUR, TIME, SERV, ANAME, UTC , ULC,
SCLAS , MJCOM,
      MONOR, MAJOR, REVAL , UDC, LNAME , COAFF,
BUPDATE, TPSN
 FROM SORTSM_BIDES
go
eof
;;
```

F-2.3.5 Sybase Constraints

Database constraints are not created when the table is created, but later in the install process. This allows the table creation to occur in any order. When the install process includes a base load of data it can be done faster if the primary keys and indexes are

created after the data is loaded. Constraints are also used to bind a rule to columns or custom data types in Sybase tables. The rule must be defined before it can be bound.

The following script excerpts create primary key constraints in Sybase.

```
isql -UDBSORT -${DBO_PWD} <<eof
ALTER TABLE SORTSM_BIDES ADD PRIMARY KEY (UIC)
go
sp_primarykey SORTSM_BIDES, UIC
go
sp_bind rule UIC_RULE, “SORTSM_BIDES.UIC”
eof
;;
```

F-2.3.6 Sybase Indexes

Indexes must be created after the database tables have been created. A Sybase index should specify a “Sybase segment name.” Clustered indexes should be used unless the table contains dynamic data. Common keys can also be used in Sybase to relate non-primary keys across tables.

The following script excerpts create database indexes in Sybase.

```
CREATE INDEX SORTSM_BIDES_LNAME
     ON SORTSM_BIDES (LNAME)
     ON DBSORT_INDEX
go
eof
;
```

F-2.3.7 Sybase Grants

Grants assign privileges on objects to database roles. In Sybase the following object-level privileges are available:

- **Select:** The object may be queried
- **Insert:** Records may be added to the object
- **Update:** Existing records in the object may be changed
- **Delete:** Existing records in the object may be deleted
- **Reference:** A table may be used (referenced) to validate a foreign key or domain key constraint

The following script excerpt performs grants in Sybase.

```
GRANT DELETE, INSERT,SELECT, UPDATE ON SORTSM_BIDES to
DBSORT_RW
go
GRANT SELECT ON SORTSM_BIDES to DBSORT_RO
go
eof
;;
```

F-2.3.8 Sybase Database Groups

Sybase groups are used to manage user privileges and database access. A group is a collection of privileges. Users within a database can only belong to one group at a time. The group definitions must be mutually exclusive. The use of 'guest' or 'visitor' groups is prohibited. Developers may not grant privileges to 'public.'

The following script excerpt provides examples for the creation of Sybase groups.

```
isql -UDBSORT -PDBSORT -S${DSQUERY} <<eof
sp_addgroup DBSORT_RO
go
sp_addgroup DBSORT_RW
go
eof
;;
```

F-2.4 Sybase Environment Variables

Key environment variables used by the Sybase system are defined during the installation of the Sybase COTS RDBMS Server segment. These environment variables specify the root of the Sybase installation tree and identify the SQL Server network connection. The environment variable definitions set by the Sybase COTS RDBMS Server segment are as follows:

```
SYBASE          /h/COTS/SYBASE
path            ($path:${SYBASE}/bin)
```

F-2.5 COE Tool Considerations

Some of the COE tools require special considerations when Sybase is used. This subsection lists the special considerations by tool.

F-2.5.1 COECreateDS

For a Sybase implementation, `COECreateDS` creates the database, the "Sybase segments," and the database owner account. The database name is set to that of '`dbo_name`.' The database owner account created by this service has an initial password

equal to the account name. The DBO account is configured to default to the created database.

F-2.5.2 COEStartDBServer

For a Sybase database server `COEStartDBServer` will check to ensure that all disks that have database files are online before starting up the server.

F-2.5.3 COEExtendDS

For a Sybase implementation, `COEExtendDS` creates the necessary data segment and adds additional storage to a Sybase database. If the new storage requirements can be serviced then the database is altered by allocating the disk storage to the database.

F-3. Informix RDBMS

This section contains information specific to the Informix Database Server and Client segments and to database segments written for the Informix RDBMS. It also provides information for developers of application segments that will access Informix Databases.

F-3.1 Informix Server Segment

The Informix COTS RDBMS Server segment contains the software required to install and maintain an instance and a database, and the software to support both client/server and server/server communications across the network. Installation of the Informix COTS RDBMS Server segment defines a common environment that provides a central means of updating all user accounts with regard to database changes. Installation of this segment also creates the database files, tablespaces, log files, temporary storage and internal data structures that make up a core database to be expanded and populated by other segments.

The core database is installed with an `informix` account that is the owner of both the core database and the database administrator. Security measures mandate that the password of the `informix` account be changed upon successful installation of this segment. The use of scripts or applications containing hardcoded passwords for these accounts, or any other Informix database accounts, is prohibited.

The Informix COTS RDBMS server segment also sets up the disk storage environment for the creation of database files. The Informix COTS RDBMS server segment installs the initial data segment in the `/opt/informix/DBS_files` directory. Other database segments may place requisite disk storage on other disk drives. The DBA services will provide specific tools to add disk storage for the database.

There are two segments associated with the Informix COTS RDBMS server segment.

1. A development segment that includes Informix OnLine Dynamic Server version 7.1x Development.

2. A runtime segment that includes Informix OnLine Dynamic Server version 7.1x Runtime.

F-3.2 Informix Client Segment

The Informix COTS RDBMS Client segment contains the Informix RDBMS executables for accessing the database, the tools and utilities that support data storage and retrieval and software to support client/server communications across the network.

There are two segments associated with the Informix COTS RDBMS client segment.

1. A development segment that includes Informix ESQL/C version 7.1x Development and Informix -SQL version 6.x Development.

2. A runtime segment that includes Informix ESQL/C version 7.1x Runtime and Informix -SQL version 6.x Runtime.

F-3.3 Informix Application Database Segments

There are some special considerations for mission-application segments that use Informix services.

F-3.3.1 DBS_files Directory

Data store segments are required to create their own dbspaces for an Informix implementation, their own tablespaces for an ORACLE implementation or their own "Sybase segments" for a Sybase implementation.

The functional storage allocation of a Data Store is implemented in Informix from within the DBMS as one or more 'dbspaces' that storing objects. Each segment must create their own database. Segments are prohibited from storing tables and indexes in another segment's dbspace or the system (root) dbspace.

F-3.3.2 Informix Tables

Scripts to create Informix tables must specify their size. The tables must be created in dbspaces created by the segment. When creating a table, the Informix 'in' parameter must be specified to direct storage allocation. The Informix LOCK MODE parameter must be specified to specify row-level locking. The following script excerpt creates a database table in Informix.

```
CREATE_TABLE)

dbaccess DBSORT <<eof

     create table SORTSM_BIDES (
     UIC        VARCHAR(6)      NOT NULL
     ,SECUR     VARCHAR(2)      NOT NULL
     ,TIME      DATE            NOT NULL
     ,SERV      VARCHAR(1)      NULL
     ,ANAME     VARCHAR(30)     NULL
     ,UTC       VARCHAR(5)      NULL
     ,ULC       VARCHAR(3)      NULL
     ,SCLAS     VARCHAR(2)      NULL
     ,MJCOM     VARCHAR(6)      NULL
     ,MONOR     VARCHAR(6)      NULL|
     ,MAJOR     VARCHAR(1)      NULL
     ,REVAL     VARCHAR(1)      NULL
     ,UDC       VARCHAR(1)      NULL
     ,LNAME     VARCHAR(55)     NULL
     ,COAFF     VARCHAR(2)      NULL
     ,BUPDATE   DATE            NULL
     ,TPSN      VARCHAR(7)      NULL)
```

```
        IN DBSPACE2 LOCKMODE ROW EXTENT SIZE 2000 NEXT
        1000
        eof
;;
```

F-3.3.3 Informix Views

Views should be created after all tables have been created to guarantee that the referenced table objects exist. A segment may create views to support legacy structures as well as application-specific database requirements. The following example script creates a database view in Informix to support a change in a column name from SECURITY to SECUR.

```
CREATE_VIEW)
     dbaccess DBSORT<<eof
     CREATE VIEW SORTSM_BIDES_1_1
          (UIC, SECURITY, TIME, SERV, ANAME, UTC, ULC,
          SCLAS, MJCOM, MONOR, MAJOR, REVAL, UDC, NAME,
          OAFF, BUPDATE, TPSN)
     AS SELECT UIC, SECUR, TIME, SERV, ANAME, UTC, ULC,
          SCLAS, MJCOM, MONOR, MAJOR, REVAL, UDC, NAME,
          COAFF, BUPDATE, TPSN
     FROM SORTSM_BIDES;
     eof
     ;;
```

F-3.3.4 Informix Constraints

Database constraints are not generally created when the table is created, but later in the install process. This allows table creation to occur in any order. It also speeds data loading if the primary keys, indexes, and other constraints are created after the data is loaded.

The following script excerpt creates a primary key constraint in Informix.

```
CREATE_CONSTRAINTS)

dbaccess DBSORT<<eof
ALTER TABLE SORTSM_BIDES ADD CONSTRAINT PRIMARY
KEY(UIC)
eof
;;
```

Similarly, database triggers are installed after all the database procedures are installed. A trigger's body may be coded completely in Informix-SQL, or it could invoke stored procedures to perform the same functions. The use of stored procedures is recommended for performance and maintainability.

F-3.3.5 Informix Indexes

An Informix index should specify a dbspace. The following script excerpt creates an Informix index.

```
CREATE_INDEX)
     dbaccess DBSORT<<eof
CREATE INDEX SORTSM_BIDES_LNAME
ON SORTSM_BIDES(LNAME)
IN dbspace4
eof
;;
```

F-3.3.6 Informix Grants

Grants assign privileges on objects to database roles.

In Informix, data access to PUBLIC is prohibited to prevent inadvertent access provided by the default PUBLIC role.

The following script excerpt performs grants in Informix.

```
ASSIGN_GRANTS)

dbaccess DBSORTS<<eof

GRANT DELETE, INSERT, SELECT, UPDATE ON
DBSORT.SORTSM_BIDES TO DBSORT_RW;
GRANT SELECT ON DBSORT.SORTSM_BIDES TO DBSORT_RO;
eof
;;
```

F-3.3.7 Informix Roles

In Informix, like Oracle, a role is a set of privileges that can be granted to users or other roles. An Informix user can take on many roles.

```
CREATE_ROLE)

dbaccess DBSORT<<eof
create role DBSORT_RO;
create role DBSORT_RW;
eof
;;
```

F-3.4 Informix Environment Variables

Key environmental variables used by the Informix system are defined during the installation of the Informix COTS RDBMS Server segment. These environment variables

set a common environment for all users, define the Informix system home directory, and uniquely identify the Informix database. The following variables are defined during the installation of the Informix Server segments.

```
INFORMIXDIR          /h/COTS/INFORMIX
INFORMIXSERVER
ONCONFIG
path                 ($PATH:${INFORMIXDIR}/bin)
```

F-3.5 Database Synonyms

A database synonym is functionality provided by both Informix and ORACLE. In Informix, a synonym is an alternate name for a table or a view. Use of synonyms should be avoided. Applications should reference objects using full object specifications (owner.table.column).

F-4. Reserved Words

Following is the current list of combined reserved words from Sybase, Informix, and Oracle, which cannot be used (irrespective of case) in naming database elements, etc.

abort
absolute
access
action
add
after
alias
all
allocate
alter
and
any
append
apt
are
arith_overflow
as
asc
assertion
async
at
audit
authorization
avg
backtab
before
begin
bell
between
binary
bit
bit_length
boolean
both
breadth
break
browse
bulk
by
call
callextern
callform
callreport
cancelform
cascade
cascaded
case
cast
catalog
channel
char
character
character_length
charindex
char_convert
char_length
check
checkpoint
close
closesql
cluster
clustered
coalesce
collate
collation
column
comment
commit
completion
compress
compute
confirm
connect
connection
constraint
constraints
continue
controlrow
convert
corresponding
count
create
cross
curindex
current
current_date
current_time
current_timesta
mp
current_user
cursor
cursqlindex
cycle
data
database
datalength
data_pgs
date
datename
datepart
datetime
day
dbcc
deallocate
dec
decimal
declare
default
deferrable
deferred
define
delete
depth
desc
describe
descriptor
diagnostics
dictionary
disconnect
disk
distinct
domain
double
drop
dummy
dump
each
else
elseif
end
end-exec
endtran
enter
entry
equals
errlvl
errorexit
escape
except
exception
exclusive
exec
execute
exists
exit
exitform
exp
external
extract
false
fetch
fetchsql
field
file
fillfactor
first
float
for
foreach
foreign
form
found
from
full
general
get
global
go

goto
grant
group
having
hidden
holdlock
hour
identified
identity
identity_insert
if
ignore
image
immediate
in
increment
index
indicator
initial
initially
inner
input
insensitive
insert
int
integer
interruptsql
intersect
interval
into
is
isolation
join
key
kill
language
last
leading
leave
left
less
level
like
limit
lineno
list
load

local
lock
log
long
loop
lower
match
max
maxextents
mchoice
menu
menubar
min
minus
minute
mirror
mirrorexit
mode
modify
module
money
month
names
national
natural
nchar
new
next
nextquery
no
noaudit
nocompress
noholdlock
nomsg
nonclustered
none
not
nowait
null
nullif
number
numeric
numeric_truncati
on
object
octect_length
of

off
offline
offsets
oid
old
on
once
online
only
open
opensql
operation
operators
option
or
order
others
outer
output
over
overlaps
pad
parameters
parentname
partial
pctfree
pendant
perform
perm
permanent
plan
position
positionform
post
precision
preorder
prepare
preserve
primary
print
printform
prior
private
privileges
proc
procedure
processexit

protected
public
raiserror
raw
rchoice
read
readtext
real
reconfigure
recursive
ref
references
referencing
relative
remote
rename
replace
reserved_pgs
reset
resignal
resource
restrict
return
returns
revoke
right
role
rollback
routine
row
rowcnt
rowcount
rowid
rowlabel
rownum
rows
rule
save
savepoint
schema
schoice
scroll
search
second
Section
select
sensitive

sequence
session
session_user
set
setuser
share
shared
shutdown
signal
similar
size
smallint
some
space
sql
sqlbegin
sqlcode
sqlend
sqlerror
sqlexception
sqlexpr
sqlrow
sqlstate
start
statistics
stripe
structure
submit
substring
successful
sum
switch
switchend
syb_identity
syb_restree
synonym
sysdate
system
system_user
tab
table
temp
temporary
test
text
textport
textsize
then
there
time
timestamp
timezone_hour
timezone_minut
e
tinyint
to
trace
trailing
tran
transaction
transfer
translate
translation
trigger
trim
true
truncate
tsequal
type
uid
under
union
unique
unknown
update
upper
usage
used_pgs
useform
user
user_option
using
validate
value
values
varchar
varchar2
variable
varying
view
virtual
visible
wait
waitfor
when
whenever
where
while
with
without
work
write
writetext
year
zone

F-5. Data Type Compatibility

Developers should use DBMS-provided data types that conform to ANSI/ISO SQL standards and are common across the different COTS DBMS This ensures consistency and portability of database objects and their elements.

Developers shall not use machine-dependent data types (e.g., float) as their behavior across different host computers cannot be predicted. Table F-3 provides a cross reference of Informix, Oracle, and Sybase data types.

Sybase	Oracle	Informix	Range
Integers:			
smallint	smallint	smallint	32,767 to -32,768
int	int	int	2^{31}-1 to -2^{31}
Decimals:			
numeric(p,s)	number(p,s)	numeric(p,s)	10^{38}-1 to -10^{38}
decimal(p,s)	number(p,s)	decimal(p,s)	10^{38}-1 to -10^{38}
float(p) *	float(b)	float(p)	machine-dependent
double precision	double precision	double precision	machine-dependent
real	real	real	machine-dependent
Date/time:			
datetime	date	datetime	valid to seconds only
Character:			
char(n)	char(n)	char(n)	255 chars or less, fixed length
varchar(n)	varchar(n)/varchar2(n)	varchar(n)	255 chars or less, variable length
nchar	nchar	nchar(n)	255 chars or less, fixed length
nvarchar	nvarchar	nvarchar(n)	255 chars or less, variable length
text(n)	long(n)	text	2^{31}-1 chars or less
Binary:			
image	raw	Byte	2^{31}-1 bytes

Table F-3: DBMS Data Types

This page is intentionally blank.